Passage Retrieval in Log Files: An Approach Based on Query Enrichment

Hassan Saneifar, Stéphane Bonniol, Anne Laurent, Pascal Poncelet, Mathieu Roche

Passage Retrieval in Log Files: An Approach Based on Query Enrichment

Hassan Saneifar[1,2], Stéphane Bonniol[2], Anne Laurent[1], Pascal Poncelet[1], and Mathieu Roche[1]

[1] LIRMM - Univ. Montpellier 2 - CNRS, France
[2] Satin IP Technologies, France
{saneifar,laurent,poncelet,mroche}@lirmm.fr
{stephane.bonniol}@satin-ip.com
http://www.lirmm.fr/~{saneifar,laurent,poncelet,mroche}

Abstract. The question answering systems are considered the next generation of search engines. This paper focuses on the first step of this process, which is to search for relevant passages containing answers. Passage Retrieval, can be difficult because of the complexity of data, log files in our case. Our contribution is based on the enrichment of queries by using a *learning method* and a *novel term weighting* function. This original term weighting function, used within the enrichment process, aims to assign a weight to terms according to their relatedness to the context of answers. Experiments conducted on *real data* show that our protocol of primitive query enrichment make it possible to retrieve relevant passages.

1 Introduction

Information Retrieval (IR) aims to find documents related to a topic specified by a user. The topic is normally expressed as a list of specific terms. However, the needs of some application domains make Information Retrieval (IR) methods inefficient. Indeed, when the goal is to find specific and concise answers, Information Retrieval systems are not relevant according to the considerable amount of documents that they retrieve as possibilities. Moreover, the information found in retrieved documents is not always correlated with the given query. That is why Question Answering (QA) systems are an important research topic nowadays. Question Answering systems aim to find a relevant fragment of a document which could be regarded as the best possible concise answer to a question given by user. There are two main categories of QA systems: (1) Open domain and (2) Restricted domain.

In the first case, questions arise about general domains and sources of information are large corpora consisting of documents of several fields (e.g. corpus of web pages). The evaluation of open domain QA systems has been built since 1999 in TREC[3] (Text REtrieval Conference) American evaluation campaigns.

[3] http ://trec.nist.gov/

In the second category, QA systems are designed to answer questions in a specific area. In this kind of QA systems, information resources are technical and specialized documents. The restricted domains, called also closed domains, have certain characteristics which make the methods of open domain QA become less useful [3]. We detail these characteristics and some features of specialized textual data which make answer retrieval more difficult in Sect. 2.

The Passage Retrieval represents an important phase of QA process. To give an efficient and reliable definition, a passage is defined as a fixed-length sequence of words which can begins and ends anywhere in a document [9]. The Passage Retrieval is the task of searching for passages which may contain the answer to a given question.

In this paper, we present our work on passage retrieval in a specialized domain. We deal with a particular type of complex textual data which are log files generated by Integrated Circuit (IC) design tools. These log files are digital reports on configurations, conditions, and states of systems. In this area, checking the quality of products requires to answer some technical and specialized questions. At this stage, our goal is to find segments of the logs that contain the answers to the questions of quality check. The particularity of such textual data (*i.e. log files*) and characteristics of restricted (closed) domains impact significantly the accuracy and performance of passage retrieval in this context.

We propose in this paper, a passage retrieval system based on a new approach of query enrichment. Our query enrichment process is based on a *learning approach* and a *new weighting function* which gives a score to terms of corpus according to their *relatedness* to the *context of answers*. The enrichment process takes place in two phases. First, we propose a method for learning the context of questions, based on the notion of "lexical world". In the learning phase we identify the terms representing the context of questions. Then, the initial queries are enriched by these terms. Secondly, we propose an *original term weighting function* which aims at giving a high score to terms of corpus which have a significant probability to exist in the relevant passages. The terms having the highest scores are included in the query in order to improve its enrichment. Our approach is an interactive system based on relevant feedback. We show that our approach gives *satisfactory* results on *real data*.

In Section 2, we present the specific characteristics of log files and also the limits of QA systems in restricted domains. Existing work concerning the QA systems are presented in Sect. 3. Section 4 presents some notions used in enrichment processes and also the first phase of query enrichment. In Section 5 we develop our approach of passage retrieval and query enrichment by presenting our novel term weighting function. Experiments on real data are presented in Sect. 6.

2 Problem Study

Log files generated by Integrated Circuits design tools are not systematically exploited in an efficient way despite the fact that they contain the essential infor-

mation to evaluate the design quality. The particular characteristics of logs, described below, make classical techniques of Natural Language Processing (NLP) and Information Retrieval irrelevant.

2.1 Information Retrieval & Log files

We consider log files as a kind of "complex textual data", i.e. containing *multi-source*, *heterogeneous*, and *multi-format* data. Indeed, in the design of Integrated Circuits, different design tools can be used in the same time while each tool generates its own log files. Therefore, despite the fact that the logs of the same design level contain the *same* information, their *structures* and *vocabulary* can vary significantly *depending* on the *used design tool*. More precisely, in order to report the same information, each design tool uses its own vocabulary.

In this domain the questions (queries) are expressed using a vocabulary that does not necessarily correspond to the vocabulary of all tools. However, a system should be able to answer the questions regardless of the type of tools that generated the log files. We explain this issue with an example. Consider the sentence "`Capture the total fixed STD cell`" as a given question (query). We produce two log files (eg., log "A" and log "B") by two different tools. The answer to the question, in log "A", is expressed as follows:

```
Std cell area: 77346 sites
non-fixed:74974 fixed:2372
```

While the answer, in log "B", is expressed in this line:

```
preplaced standard cell is: 24678
```

As shown above, the same information in two log files produced by two different tools, is represented by different structures and vocabulary. The keywords of the question (i.e. `Fixed, STD` & `cell`) exist in the answer extracted from log "A" while the answer from log "B" contains only the word "`cell`". Insofar as there is a dictionary associating the word "`STD`" with "`standard`", we can also consider the word "`standard`". However, by giving these two words as a query to an information retrieval system, irrelevant passages of log "B" are retrieved:

```
standard cell seeds is : 4567
Total standard cell length = 0.4536
```

This can be explained by the fact that the question is expressed using the vocabulary of log "A" which is different from the vocabulary of log "B". In other words, for a given question, the *relevant answers* found in the logs of *some tools* do *not* necessarily contain the *keywords* of the question. Therefore, the initial query (*created by taking the keywords of question*) may be relevant to logs generated by a tool, but irrelevant to logs generated by another tool[4] whereas we aim to answer questions regardless type of tools generating log files.

The *existence of question keywords* (or their syntactic variants) in a *passage* is an important factor to assess the *relevance* of the passage. The approaches which are based on the notion of common terms between questions and passages are detailed in Sect. 3.

[4] While all of these logs report the same information using different vocabularies

Moreover, the performance of a QA system depends largely on redundant occurrences of answers in the corpus in which answers are seek [1][7]. The methods developed for QA systems are generally based on the assumption that there are several instances of answers in corpus. But information is rarely repeated in the log files of IC design tools. This means that for a question, there is only one occurrence of the answer in the corpus and thus one relevant passage (containing the answer).

In addition, design tools change over time, often unexpectedly. Therefore, the *format of the data* in the log files changes, which makes automatic data management difficult. Moreover, the language used in these logs is a difficulty that impacts information extraction methods. Although the language used in these logs is English, their contents do not usually comply with "*classic*" grammar. In the processing of log files, we also deal with multi-format data: textual data, numerical data, alphanumerical, and structured data (e.g., table and data block). There are also many technical words that contain special characters which are only understandable considering the domain documentation. Due to these specific characteristics of log files, NLP and IR methods, developed for texts written in natural language, are not necessarily well adapted to log files.

We therefore suggest the enrichment of initial queries in order to make them relevant to all types of logs (*generated by any kind of design tools*). We explain the query enrichment process in Sect. 4

2.2 Passage Retrieval in Log Files

The passages retrieval phase influences significantly the performance of QA systems because final answers are sought in the retrieved passages. Most QA systems for a given question, extract a large number of passages which likely contain the answer. But an important point in QA systems is to limit, as much as possible, the number of passages in which the final answer extraction is performed. Since we are situated in a very specialized domain, high precision in the final answers (i.e. the percentage of correct answers) is a very important issue. This implies that the passage retrieval system has to classify relevant passages (based on a relevance score) in the top positions among all retrieved candidate passages.

3 Related Work

Most passage retrieval algorithms depend on occurrences of query keywords in corpus [9]. To enhance the query, the use of morphological and semantic variants of query keywords is studied in [2].

The reformulation of questions (also referred to as *surface patterns* and *paraphrases*) is a standard method used to improve the performance of QA. The technique is based on identifying various ways of expressing an answer given a natural language question [5]. For example, for a question like "*Who founded the American Red Cross?*", QA systems based on surface patterns seek reformulations like "*the founder of the American Red Cross is X*" or "*X, the founder of*

the American Red Cross". The question reformulation using surface patterns is also exploited in TREC9 and TREC10. [8] and [5] present different approaches to take semantic variations (semantic reformulations) into account in order to complement the syntactic variants. To find relevant passages, [6] evaluates each passage using a scoring function based on the coverage of "question keywords" which exist also in the passage. QA systems also use query expansion methods to improve performance of retrieval. These methods can use the thesaurus [4] or be based on the incorporation of the most frequent terms in the m relevant documents.

Despite the satisfactory results achieved by the use of surface patterns and syntactic variants in mentioned work, these methods are irrelevant in the context of log files according to the problems described in Section 2. Indeed, the main reasons for irrelevancy of such methods are related to the fact that an answer is not reformulated in different ways in a corpus of log files. Also, there is a lack of redundancy of answers in corpus of logs. In addition, there are several technical and alphanumeric keywords in the domain of log files for which the use of syntactic or semantic variants appears to be complex or unmeaning.

4 Passage Retrieval Based on Query Enrichment

We propose in this paper, a passage retrieval approach based on a new interactive process of query enrichment. The enrichment of query is based on a context learning process and is associated with a novel and original term weighting function. Our protocol of context learning is designed to determine the context of a given question by analyzing the terms [5] co-occurring around the question keywords in the corpus. The new term scoring function proposed in this paper, identifies the terms which are related to the answers.
The architecture of our approach consists of three main modules:

1. Enrichment of the initial query by context learning
2. Enrichment by terms which are likely related to answer
3. Passage retrieval using the enriched query

The first module enriches the initial query extracted from a question in natural language. This module aims at making the initial query relevant to all types of logs which are generated by different tools. At this step, by learning the context of question, we enrich the initial query by the most significant terms of the context.

The second module is activated for a second enrichment of the query in order to obtain a higher accuracy. At this phase, we aim at identifying the terms which are likely related to answer in order to integrate them in the query. For this purpose, we propose a process of scoring of terms based on a new weighting function. The weighting function is designed to give a score to terms according to their relatedness to answers. We devote Section 5 to this topic.

[5] In this paper, the word "term" refers to both words and multi-word terms of the domain

In the third module, we seek the relevant passages in the logs generated by a tool different from the one which has been used in the learning phase. That is, we have two different corpora of logs. The first one (called *training corpus*) is used in learning phase and the second corpus (called *test corpus*) is the corpus in which we retrieve the passages relevant to given questions. The logs of the test corpus have structures and a vocabulary significantly different from the logs of the training corpus. In our approach, we look for specialized context (called hereafter "*lexical world*") of question keywords. In order to characterize and present in a relevant way the specialized context of keywords, we use the terminological knowledge extracted from logs. Before explaining the query enrichment processes, we develop the concept of "lexical world" and the use of terminological knowledge in the following subsections.

4.1 Lexical World

The lexical world of a term is a small fragment of document in which the term is seen. We consider this fragment specialized context of the term because terms located around a term (within a small fragment) generally have a strong semantic and/or contextual relations. Therefore, by determining the *lexical world* of a term in a document, we identify *terms* that *tend* to *appear around it* in that document. We do not put any limit on the size of lexical worlds (eg., a few lines, a few words, etc.) as it have to be determined pragmatically based on the type of documents.
In order to present the lexical world of a term, several solutions are possible. As a first solution, we characterize the lexical world of a term by a set of "words" (*called also bag of words*) which are located around the term and present "Noun", "Verb", or "Adjective" parts of speech. As a second solution, the lexical world of a term is presented by co-occurring words like bigrams of words (i.e. any two adjacent words) or multi-word terms (few adjacent words forming a meaningful term) which are seen around the term. We detail this point in the next section.

4.2 Terminological Knowledge

As mentioned above, the lexical worlds can be characterized in different ways: By "words", "multi-word terms", or "bigrams of words". According to our experiments, the multi-word terms and words are more representative than bigrams of words. Hence, we create two types of lexical world: (1) Consisting of words and (2) Consisting of multi-word terms and words. In order to determine the multi-word terms, we extract the terminology of logs using the method presented in [13]. This method adapted to the specific characteristics of logs, extracts the multi-word terms according to syntactic patterns in the log files. To choose the relevant and domain-specific terms, we use the terminology validation and filtering protocol presented in [12]. We have finally the valid and relevant multi-word terms to characterize lexical worlds.

4.3 Query Enrichment by Context Learning

We explain here the first module of our query enrichment approach. For a given question, we look initially to learn the context of the question and characterize it by its most significant terms. These terms represent at best the context of the question. Thus, it is expected that the passages corresponding to the question share some of these terms regardless of different kind of log files.
Firstly, we seek lexical worlds of question keywords. The found lexical worlds and the initial query are vectorized. We use an IR system based on Vector Space (VS) model in order to select the lexical world most correlated to the initial query. Then, we choose the most representative terms of selected lexical world. For this purpose, we select the n terms having the highest tf-idf scores [11]. We get the first enriched query by inserting the selected terms in the initial one.
Since the next phase of query enrichment based on the novel weighting function is the main contribution of this paper, we develop it in Section 5. Thus, we explain in the following subsection how we look for relevant passages once initial queries are enriched.

4.4 Passage Retrieval in Log Files

We detail here the process of finding relevant passages on the test corpus of log files. First, we segment the logs of the test corpus. Segmentation is performed according to the structure of the text like data blocks, tables, separating lines, etc. Each segment is seen as a passage of log files containing potentially the answer. Second, we enrich the initial query in order to make it relevant to all types of log files. We remind that we aim at adapting the initial query to vocabulary of all types of log files by our query enrichment approach. Third, we build a system of IR in order to find the relevant passages. The IR system uses an indexing function and a similarity measure. In this phase, we experiment our IR system by using *tf-idf* and Binary representation as indexing functions. The Cosine and Jaccard measures are used as a similarity measure. The IR system gives a relevancy score to every passage (segment) according to the enriched queries. Then, we order the passages based on their relevancy score and propose the top-ranked passages to the user.

Several approaches of passage retrieval return a considerable number of candidate passages. Our experiments conducted on real data assert that in more than 80% of the cases, the relevant passage is located among the three top-ranked passages. That is, our approach often ranks the relevant passage among the three top candidate passages returned by a system as possible results.

5 How to Find Terms Correlated to Answers

To improve the relevance of the query to different types of logs, we propose a second module of enrichment of the query. In this module we have as input the query enriched in the phase of context learning and the test corpus of logs in

which we seek the relevant passages (different from the training corpus which is used in the context learning phase).

Our motivation is to find the terms in the logs of the *test corpus*, which are likely to exist in the relevant passage and are, therefore, related to the answer. For this purpose, we offer here a term selection process based on a new term weighting (scoring) function. Terms will be selected based on their score obtained by this scoring function. This function gives a score to each term based on three assumptions:

- the most *correlated terms* to the answer are in a *lexical world* representing the *context* of the question
- the final query must contain *discriminant terms* (i.e. terms which do not exist in several passages)
- *most of the terms of a relevant query* are associated with the corresponding *relevant passage* (i.e. the relevant passage contains most of query keywords)

Based on the first assumption, for each query keyword, we extract their lexical worlds in the *test corpus*. We note that the system has no information on the logs of the test corpus and relevant passages[6]. We finally obtain a set of lexical worlds, corresponding to the query keywords, which are extracted from the test corpus. Our scoring function is based on the two last assumptions.

Firstly, we seek to select discriminative terms (i.e. terms that do not exist in several lexical worlds). In other words, we look for terms with a very low frequency of occurrence in different lexical worlds. For this, we use the *idf* (inverse document frequency) function by considering each lexical world extracted in the previous step as a document and all lexical worlds as the corpus. In this way, we favour terms which exist in one or a very small number of lexical worlds.

Secondly, in order to favour the terms which likely exist in the relevant passage, we give another score (*besides idf*) to each term of lexical worlds based on the third assumption. For a given term T, this score that we call LWF (Lexical World Frequency), depends on the number of query keywords which are associated with the lexical world to which the term T belongs. This score presents a form of *df* (document frequency) where a document corresponds to all of lexical worlds associated with a query keyword. Indeed, by this score, we measure how important is the lexical world in which the given term is located. This importance is calculated according to the number of query keywords which are associated with the lexical world. The LWF formula for the term i existing in the lexical world j is calculated as follows:

$$\mathbf{lwf_{ij}} = \mathbf{1}/\log(\mathbf{M}/\mathbf{n_j})$$

M is the total number of query keywords and n_j shows the number of query keywords which are associated with the lexical world j.

[6] The learning process is performed on the logs of the training corpus which are generated by a tool using a significantly different vocabulary and structures from the tool generating the logs of the test corpus.

The final score of a term that we call TRQ (Term Relatedness to Query) is calculated using the following formula:

$$\mathbf{TRQ} = \alpha * (\mathbf{lwf}) + (\mathbf{1} - \alpha) * \mathbf{idf}$$

According to the experiments, the most relevant value of α is 0.25[7]. This means that we give more weight to the frequency of terms in the corpus and a smaller weight but which influences the final results to lwf.

We explain with an example, the process of selection of terms which are likely related to the answer. Supposing Q=$\{W_a, W_b, W_d\}$ as a query enriched by the first module (learning phase) and $\log_b$ as a log file containing seven segments:
$\mathbf{S_1}$=$\{W_a\ W_k\ W_m\ W_b\}$ $\mathbf{S_4}$=$\{W_a\ W_c\ W_e\ W_q\}$ $\mathbf{S_7}$=$\{W_b\ W_c\ W_k\}$
$\mathbf{S_2}$=$\{W_d\ W_k\}$ $\mathbf{S_5}$=$\{W_b\ W_e\}$
$\mathbf{S_3}$=$\{W_z\}$ $\mathbf{S_6}$=$\{W_z\}$

We consider that the border of lexical worlds (*border of the selected fragment of text around a given term*) corresponds to the border of segments (i.e. a lexical world is not bigger than the corresponding segment). Among sept segments, there are fives which are associated with terms of Q. Thus, we obtain S_1, S_2, S_4, S_5, S_7 as the set of lexical worlds of question keywords. The following lists shows the lexical worlds associated to each keywords of the question[8].
W_a:$\{S_1, S_4\}$ W_b:$\{S_1, S_5, S_7\}$ W_d:$\{S_2\}$
Here, for instance, the word W_a in the query Q is associated with two lexical worlds (S_1 and S_4). The *idf* score of the word W_k, for example, is equal to $log(5/3) = 0.22$ because, the word W_k exists in three lexical worlds (S_1, S_2 and S_7) among fives. The value of lwf for the word W_k in the segment S_1 is calculated as following: $lwf_{2,1} = 1/\log(3/2) = 5.8$. Indeed, there are two words in the query Q (i.e. W_a and W_b) which are associated with the lexical world S_1 (the lexical world in which the word W_k is located). We note that for a given term, the value of lwf depends on the lexical world in which the given term is located. For example, the value of lwf for the word W_k located in segment S_2 is equal to $lwf_{2,2} = 1/\log(3/1) = 2.1$ as there is just one keyword of the query Q associated to S_2. This means that W_k located in the segment S_2 is less significant (less related to the query) than when it is located in the segment S_1.

Once the TRQ score of all terms of lexical worlds are calculated, we identify the k *highest scores* and select the terms having these scores. However, among the terms selected based on their TRQ scores, there are terms having the same score. To distinguish these terms, we assess their *tendency* to appear *close* to the keywords of the initial query. In other words, we seek to calculate the *dependence* of *selected terms* to the *keywords* of question in the context. For this purpose, we choose the "Dice" measure. This statistical measure has a good performance

[7] We justify the selected value of α by presenting some results in `http://www.lirmm.fr/~saneifar/experiments/TRQ.pdf`

[8] Since a word can be used in different contexts (i.e. different fragments of document), it can be associated with several lexical worlds

for tasks of text mining [10]. For a term T and a query keyword W, we calculate the Dice measure as following:

$$Dice(T, W) = \frac{2 * \mid (T, W) \mid}{\mid T \mid + \mid W \mid}$$

For us, $\mid (T, W) \mid$, number of times T and W occur together, corresponds to the number of times where T and W are located in the same line in the corpus of logs. $\mid x \mid$ shows the total number of occurrences of x in the corpus. Finally, the value of Dice measure allows to distinguish the terms obtained in the previous step, which have equal TRQ score. The final score of these terms is obtained by the sum of Dice value and the TRQ score. Note that we select at first the terms according to their TRQ score (i.e. terms having highest TRQ score are selected). If we have the terms having the same TRQ score, we distinguish them by calculating their Dice values.

Finally, as described above, the system ranks the terms based on their TRQ scores (considering Dice values for terms having the same score). The system recommends to the user the k top-ranked terms in order to enrich the query. Our system is also able to integrate automatically the k top-ranked terms into the query in an autonomous mode. The enriched query EQ_{mod2} will be used in passage retrieval.

6 Experiments

We test the performance of our approach on a corpus of log files from the real industrial world (data from the Satin IP company). There are 26 questions which are expressed in natural language and are extracted from standard check-list. Log files are segmented according to their structures (blank lines, tables, data blocks, etc.). Each segment is potentially a relevant passage. Note that for a given question, there is only one relevant passage (*segment*) in the corpus (i.e. there is just one occurrence of the answer in the corpus). The relevance of passages is evaluated as whether the final answer is situated in the passage.

The test corpus that we use for the experiments contains 625 segments and is about 950 KB. Each segment consists of approximately 150 words. The training corpus used in the phase of learning the context of questions consists of logs reporting the same information as logs of the test corpus, but generated by a totally different tool – *thus different vocabulary and segmentations*. For a given question, initial query is obtained by taking the question keywords.

In each category, we experiment our approach while the IR system uses different indexing functions and similarity measures. In order to evaluate the performance of our approach in these different conditions, we use *Mean Reciprocal answer Rank* (MRR) used in TREC as the measure of performance evaluation [14]. This measure takes into account the rank of the correct answer (among the ranked list of candidate answers).

$$MRR = \frac{1}{nb(Question)} \sum_{i=1}^{nb(Question)} \frac{1}{rank(answer)}$$

In the experiments, we measure the performance of passage retrieval using the enriched queries. We aim at comparing the performance of passage retrieval using the enriched queries with the performance of passage retrieval using the initial queries (not enriched). That shows how our query enrichment approach improves the relevance of initial queries. Tables 1a, 1b present respectively the

	tf-idf		Binary	
	Cos	Jac	Cos	Jac
$\%P(1)$	9	8	11	8
$\%P(2)$	5	4	3	3
$\%P(3)$	4	3	2	2
MRR	0.51	0.50	**0.58**	0.48

(a)

	tf-idf		Binary	
	Cos	Jac	Cos	Jac
$\%P(1)$	20	18	19	13
$\%P(2)$	3	1	0	4
$\%P(3)$	1	3	2	1
MRR	**0.83**	0.78	0.79	0.65

(b)

Table 1: Percentage of question $\%\mathbf{P}(\mathbf{n})$ for which the relevant passage is ranked as "**n**". **(a)** performance obtained by using the ***not enriched queries*** (initial queries), **(b)** performance obtained by using the ***enriched queries***

results of the performance of passage retrieval using the not enriched query and the enriched ones. $\%P(n)$ is the *percentage of questions* for which the *relevant passage* is *ranked as* n among the retrieved candidate passages as possibilities. Here, we show the results for the three first ranks. As shown in Tab. 1a, by using the not enriched queries, we obtain a MRR value equal to 0.58 in best conditions. While by enriching the initial queries as mentioned in this paper, the MRR improves significantly and reaches 0.83 in best conditions. According to the results, in the best configuration of the IR module and by *using the enrichment of queries*, the relevant passage is ranked in 76% of cases as the first passage among the candidate passages returned by the system. Also, in 92% of cases, the relevant passage is located (ranked) among the three top-ranked passages returned by the system when there are about 650 passages in the corpus.

7 Conclusions

We have presented a process of double enrichment of initial queries in order to improve the performance of passage retrieval in log files. The heterogeneity of the vocabulary and structures of log files and the fact that the keywords used in the questions expressed in natural language do not exist necessarily in the logs make the passage retrieval difficult. Despite these characteristics, our approach makes it possible to adapt an initial query (i.e. list of question keywords) to all types of corresponding log files whatever is their vocabulary. According to the results, by our query enrichment protocol which is based on our novel weighting function called TRQ (Term Relatedness to Query), we obtained a value of MRR equal to 0.83 while the value of MRR was equal to 0.58 by using the *not* enriched queries. We plan to evaluate our system with other models of Information Retrieval.

Improving the new term weighting function used in the second phase of query enrichment, represents a major point in the future work.

References

1. Brill, E., Lin, J., Banko, M., Dumais, S., Ng, A.: Data-intensive question answering. In: In Proceedings of the Tenth Text REtrieval Conference (TREC). pp. 393–400 (2001)
2. Chalendar, G., Dalmas, T., Elkateb-Gara, F., Ferret, O., Grau, B., Hurault-Plantet, M., Illouz, G., Monceaux, L., Robba, I., Vilnat, A.: The question answering system qalc at limsi, experiments in using web and wordnet. In: TREC (2002)
3. Doan-Nguyen, H., Kosseim, L.: The problem of precision on restricted-domain question answering. In: Proceedings the ACL 2004 Workshop on Question Answering in Restricted Domains (ACL-2004). ACL- 2004, Barcelona, Spain (July 2004)
4. Jing, Y., Croft, W.B.: An association thesaurus for information retrieval. In: RIAO 94 : Computer-Assisted Information Retrieval (Recherche d'Information et ses Applications). pp. 146–160 (1994)
5. Kosseim, L., Yousefi, J.: Improving the performance of question answering with semantically equivalent answer patterns. Data Knowl. Eng. 66(1), 53–67 (2008)
6. Lamjiri, A.K., Dubuc, J., Kosseim, L., Bergler, S.: Indexing low frequency information for answering complex questions. In: RIAO 07 : 8th International Conference on Computer-Assisted Information Retrieval (Recherche d'Information et ses Applications). Carnegie Mellon University, Pittsburgh, PA, USA (2007)
7. Lin, J.: An exploration of the principles underlying redundancy-based factoid question answering. ACM Trans. Inf. Syst. 25(2), 6 (2007)
8. Mollá, D.: Learning of graph-based question answering rules. In: Proc. HLT/NAACL 2006 Workshop on Graph Algorithms for Natural Language Processing. pp. 37–44 (2006)
9. Ofoghi, B., Yearwood, J., Ghosh, R.: A semantic approach to boost passage retrieval effectiveness for question answering. In: ACSC '06: Proceedings of the 29th Australasian Computer Science Conference. pp. 95–101. Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2006)
10. Roche, M., Kodratoff, Y.: Text and Web Mining Approaches in Order to Build Specialized Ontologies. Journal of Digital Information 10(4), 6 (2009)
11. Salton, G., Buckley, C.: Term weighting approaches in automatic text retrieval. Tech. rep., Ithaca, NY, USA (1987)
12. Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., Roche, M.: Mining for relevant terms from log files. In: KDIR'09 : Proceedings of International Conference on Knowledge Discovery and Information Retrieval. Madeira, Portugal (October 2009)
13. Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., Roche, M.: Terminology extraction from log files. In: DEXA'09: Proceedings of 20th International Conference on Database and Expert Systems Applications. pp. 769–776. Lecture Notes in Computer Science, Springer (2009)
14. Voorhees, E.M.: The trec-8 question answering track report. In: In Proceedings of TREC-8. pp. 77–82 (1999)